Spyndra 1.0: An Open-Source Proprioceptive Robot for Studies in Machine Self-Awareness

Department of Mechanical Engineering, Columbia University in the City of New York

Ori Kedar[1], Christie Capper[1], Yan-Song Chen[1], Zhaoyang Chen[1], Julia Di[1], Yonah Elzora[1], Lingjian Kong[1], Yuanxia Lee[1], Julian Oks[1] Jorge Orbay[1], Fabian Stute[1], Chad Tarpley[1], Joni Mici[1], and Hod Lipson[1]

Abstract— This paper describes Spyndra, a quadruped robot created as an open source platform for studying machine self-awareness. Our key hypothesis is that a machine is self-aware to the degree it can simulate itself, and that self-simulation is essentially the ability to predict sensations from actions. In order to facilitate studies in this direction, we create the first of what we see as a series of open platforms that provide rich proprioceptive feedback. This paper provides descriptions of Spyndra's hardware and software and an analysis of two Inertial Measurement Unit (IMU) datasets to illustrate Spyndra's competency as a machine learning platform. After establishing this, we describe our methods to progress in machine learning, and thus also include data from machine learning models, an example simulation model, and a comparison between a simulated gait and real gait that allow us to begin pushing the boundaries towards what a machine can model of its own actions. These series of information can serve as a baseline for future studies.

I. Introduction

QUESTIONS over the nature of self-awareness and conscious-ness have occupied philosophers and physicians for millennia. The relatively recent advent of robotic systems, combined with machine learning technologies, has opened a new window into these age old questions. For the first time, some of the prevailing conjectures or models can be put to a test.

The key hypothesis we aim to study is the idea that consciousness, or self-awareness, is essentially the ability to self-simulate, or to perform 'mental time-travel,' and that emotions are essentially internal appraisals of that prediction.

A second aspect of our working hypothesis, is that self-awareness is not a black-or-white characteristic that creatures either possess or not. Alternatively, self-awareness lies on a continuum from machines with no self-awareness to human-level self-awareness, and beyond. If true, this hypothesis implies that we can begin experimenting with systems that might possess minute amounts of self-awareness.

Spyndra is a robotic platform that could potentially be capable of a small level of self-awareness. Utilizing min-imal proprioceptive sensation to track its internal motor commands and record its own orientation and acceleration, Spyndra can learn about its physical form. We hope that by sharing this platform and data produced by it, researchers engaged in this line of research can study self-awareness on a common platform.

Figure 1: Spyndra printed in a camouflaged brick pattern

Growth within the robotics field has traditionally been limited by barriers to entry such as expensive components and complex, inaccessible manufacturing. By contrast, Spyndra was designed as an open source platform, using off-the-shelf electronics and 3D printed, easily-assembled hard-ware, with all necessary files and instructions available on the project's website. Table 1 provides a comparison of similar low cost walking robots, both academic and hobby in origin. Spyndra is comparable in price to these alternatives, at approximately $600 to build and operate. Furthermore, by limiting the DOF to eight, Spyndra reduces the complexity of potential gaits, making machine learning a more computationally tractable problem. Calibration procedures designed specifically for the robot ensure synchronicity be-tween software and hardware and the repeatability of experiments. We have also developed control software for Spyndra, allowing gaits generated through machine learning to be seamlessly commanded to the robot, as well as a dataset of IMU readings that can serve as a baseline for further re-search. Though this paper presents Spyndra in its first version, the public availability of native files, including Python scripts and CAD files, allows for crowd-sourced customization and improvement of the system as it is adopted.

	Image	Price	Open-source hardware	Sensors	"DoF" # & type
Spyndra[12]		$575	Yes: provides STL and CAD	IMU - 1, camera	8 - rotary
Instructables Arduino Quadruped [13]		$540	Yes	Triple axis accelerometer	12 - rotary
RobotShop Lynxmotion SQ3U Symmetric Quadruped Walking Robot [14]		$550	No: assembly kit	Not listed	12 - rotary
Instructables Spider Robot [15]		~$100	Yes: provides STLs, not CAD	Not listed	12 – rotary
A 3D Printed Quadruped Robot (Instructables		~$450	Yes: provides STLs and CAD	Not listed	12 – rotary
Aracna [17]		$1,350	Yes: provides STL and CAD	Not listed	8 – rotary
Hexy – Programmable Hexapod Kit [20]		$250	Yes: provides Code, Laser Cutter DXF/STL/CAD files	Ultrasonic Distance Sensor	18 - rotary
Lynxmotion A-Pot Hexapod [19]		$1499 (No electronics)	No: assembly kit	Force Sensors	18 - rotary

TABLE 1: COMPARISON OF SPYNDRA TO SOME OTHER WALKING ROBOTS LESS THAN $1,500

II. HARDWARE

The hardware strikes a balance between being user-friendly to a wide audience and sophisticated enough to achieve a wide variety of tasks. The website includes a bill of materials, all native CAD and STL files, and instructions on how to fabricate and assemble the hardware, integrate the electronics, and implement the software. Spyndra's sensor systems include a camera and an Inertial Measurement Unit (IMU). These sensors provide information necessary for Spyndra to develop a model of itself and interact with its environment. The hardware can be customized to accommodate additional sensors.

For convention, this paper will refer to the upper section of each leg as the 'Femur,' and the lower section as the 'Tibia' as illustrated in Figure 2. A central chassis holds the Raspberry Pi 3B micro-controller, Lithium-ion battery (power source of the controller), and a Lithium polymer battery (power source for the servo motors), as well as several sensors. The chassis also houses four servos linked to the femur.

Unlike Spyndra's Creative Machines Lab predecessor, Aracna, the eight high-torque metal gear analog servo motors (Power HD 1501MG) directly drive each joint of the robot [5]. The two-pronged femurs support both sides of the motors to prevent load paths orthogonal to the motors intended axis of motion. Adhering to this design principle maximizes servo life and improves Spyndra's overall robust-ness. The motors slide effortlessly into place in both the chassis and tibia, and are connected to the femur using standard servo horns. This direct drive, as opposed to Aracna's linkage system, results in low friction/low hysteresis motion that can be more accurately represented in simulation.

Spyndra is comprised of entirely 3D-printed parts, with all necessary STL files available on the project's website. The parts are nominally designed and tested for fabrication via Fused Filament Fabrication [9] using typically low cost desktop 3D printers (Ultimaker 2 Extended+) which enables the design of topologically complex bio-inspired parts.

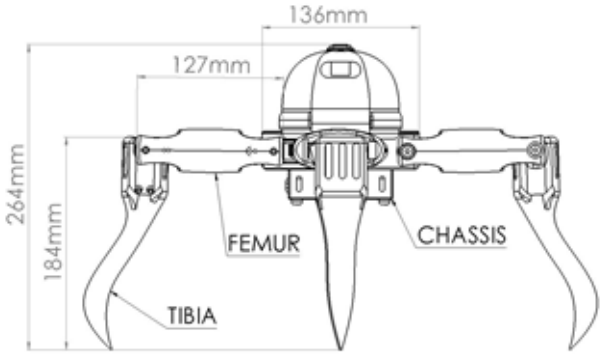


Figure 2: Labeled and dimensioned drawing of Spyndra

The recommended materials are Polylactic Acid (PLA) and ABS plastics, which are inexpensive and allow for the simple application of heat-inserts for fastening. We also fabricated Spyndra models using a Stratsys J750 printer which uses PolyJet technology [9] to achieve full color and multi-material printing, as shown in Figure 1. The organic textures on these experimental models challenge the metallic motif which we have come to accept for modern day robots such as BigDog.

Various 3D printer settings, including wall-thickness, infill density, and layer deposition orientation, have been iteratively tested. Hardware failure was initially a recurrent issue with repeated use, however incidences of fracture have been reduced, almost to entirety, through minor design revisions and enhanced printer settings. Optimized settings and print orientations for Ultimaker 2.0 3D printers can be found on the project website.

Designed for ease of assembly, the components are either pressfit or fastened using screws and heat-set inserts. No adhesives are needed in the assembly of Spyndra. The only tools necessary to assemble Spyndra are a few screw-drivers and a soldering iron.

Powered by two batteries, Spyndra can function as an untethered robot. The Raspberry Pi 3 can either be programmed to execute autonomous programs upon booting, or receive commands wirelessly via USB, Bluetooth, or SSH protocol. Spyndra's present design can run for approximately twenty minutes on one charge, but extra Lithium polymer batteries for the servo motors can be added to in-crease lifespan.

Spyndra costs around $600 to build and operate. The price breakdown can be seen in Table 3. If printed with recommended settings using PLA filament, Spyndra weighs 1.55kg. However, further lightweighting and cost reduction can be accomplished by reducing infill settings, using lower torque motors, and higher strength printing materials. These changes may come at the expense of component robustness and lifespan.

3D Printing Materials	$70
Controller	Raspberry Pi 3 + Adafruit Servo Hat: $55
Motors	8 x Power HD 1501MG: $160
Batteries	Li-Ion, LiPo, Battery Charger, Voltage Regula-tor: $115
Sensors	BNO055 IMU, Camera: $75
Misc. Electronics and Fasteners	$120

Table 3: Cost breakdown of Spyndra

To minimize mass, Spyndra uses a small Raspberry Pi-compatible camera. The camera interfaces with Spyndra's software by using the Raspicam commands native to the Raspberry Pi. The visual information from the camera, cou-pled with deep learning networks, offer a multitude of capa-bilities, including object recognition and the ability to obtain depth information. The camera has so far been used for

im-age recognition, with the ultimate aim of finding waypoints for path planning.

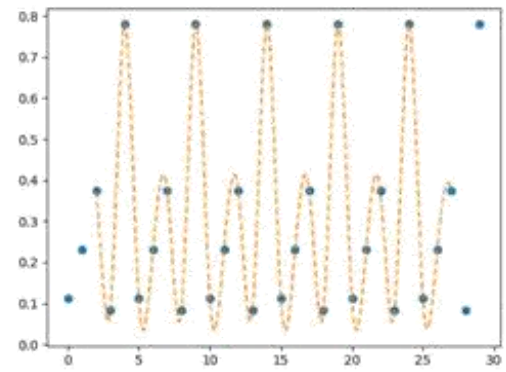
Adafruit's BNO055 is the Inertial Measurement Unit (IMU) used for measurements of acceleration, rotation, and magnetic orientation each along three dimensions and pro-vides primary feedback for gait generation. Using integra-tion, the IMU can provide a variety of information about Spyndra's physical state, such as its position, velocity, and acceleration. Since the Spyndra architecture exhibits two-way symmetry, the IMU is placed in the geometric center of the chassis so it collects the most inertially relevant data. The output from the IMU can be used to train Spyndra's algorithms to improve its ability to learn to walk.

III. Gait Generation Software

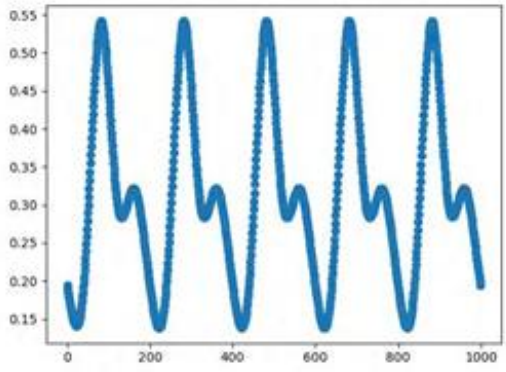
Spyndra's software suite currently consists three spline generation scripts and two runner scripts. The first spline generator is a random spline generator, which creates two random arrays of five points each, one array corresponding to femur position, and the other tibia position. The generator then fits the arrays to splines, and outputs a large array of percentages to be mapped to motor angles, see Figure 4. The five auto generated points are also tested for high deviation, and points too far away from each other are re-generated to avoid erratic movements which can damage the servo motors. This process results in a "random gait," which will be realized on the hardware using a runner script. The second spline generator produces what we call a "standing gait." It works similarly to the first but always produces the same motor angles, in which Spyndra has no translational movement, but only gyrates the chassis. This gait is meant for IMU calibration and testing. The third spline generator produces a "manual gait." The user inputs the desired number of motor coordinates for Spyndra's femur and tibia joints, and the generator fits these motor coordinates to a spline which is then outputted as an array of motor angles. For all three gait generators, the user can decide the number of times the spline is repeated.

The first runner script takes the output of any of the three generators and parses the splines, with each leg running the same spline. First, the program prompts the user for the desired phase offset, which is the amount of lag between legs as they run the spline. The spline, which consists of an array of percentages, is then mapped to the appropriate maximum and minimum motor angles (interpreted as PWM signals) which are referenced from a calibration log file. The PWM signals are sent to the servo motors while the IMU data is logged in a separate file. The second runner script is identical to the first, but accepts a time offset between legs, as opposed to a phase angle.

Before executing the gait, both runner scripts first slowly move Spyndra to a standing position with the femurs parallel and the tibias perpendicular to the robot's chassis. Once Spyndra is standing, the designated spline runs for the de-sired amount of loops. Finally, once the spline has finished running, Spyndra is moved to a sitting position where the tibia and femur are outstretched and the robot is resting on its chassis. Upon completion, any random gait generated has the option to be saved to a log file if desired.



(a)



(b)

Figure 4: Random points generated for gait are (a) fit to spline, then (b) the spline is sampled to produce the gait.

IV. Calibration

To ensure the repeatability of gaits, proper calibration of Spyndra is key. When properly calibrated, all four of Spyndra's legs will go to the same physical position when given the same PWM command, as illustrated in Figure 5. The robot consists of 8 servo motor actuators that are connected to the legs using servo horns (as discussed in Section : Hardware). The servos have a travel limit of roughly 165 degrees, and the horns must be attached in the proper orientation. To ensure that all limbs are properly calibrated, we have engineered a combined mechanical and software approach. First, the motors are unplugged and shifted all the way back to the maximum angle (counter clockwise for tibia, clockwise for femur). A mechanical jig is then placed and used to attach the horn to each legs servo motor as shown in Figure 6. This provides a rough calibration, but the coupling mechanism of the horn limits the precision of each legs orientation to within 14 degrees.

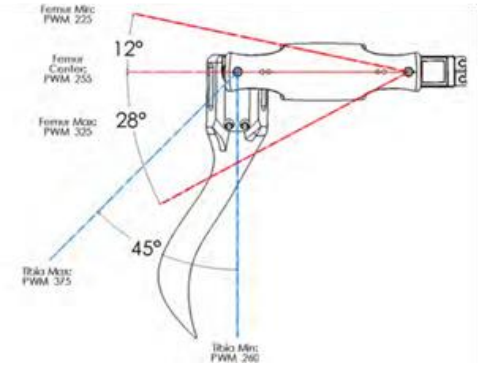


Figure 5: Range of motion of tibia and femur with corresponding PWM signals

Figure 6: The 3D printed mechanical jig (grey) is placed on the motor and guides the coupling of the servo horn (black) to the proper orientation.

Second, to further refine Spyndra's leg calibration, we lay Spyndra on a flat surface and raise its femurs and tibias to their maximum positions. Then, one at a time, each femur is lowered until it touches the flat surface. The angle at which contact occurs is recorded for each individual femur, and these angles are used to define the range of motion of each femur. The femurs are then raised ten degrees above con-tact, and the same process is done for the tibias. The lowest increment of angle is about 0.4 degrees, so all the legs are calibrated within 0.4 degrees of each other. This allows users to design gaits that can treat each leg as equivalent to the others. Figure 7 demonstrates this process.

V. IMU Dataset

To demonstrate Spyndra's suitability as a robotic gait machine learning platform, we wanted to ensure that our data is replicable by first taking two datasets using IMU measurements. These datasets demonstrate the repeatability of measurements and can serve as a baseline for future gait studies.

Both datasets consist of time, yaw, pitch, roll, x-acceleration, y-acceleration, and z-acceleration data from Spyndra. In the first, Spyndra performs the "standing gait" (described in the Section: Gait Generation Software), with a phase offset of 45 degrees. In this "standing gait," Spyndra slowly gyrates the chassis while standing stationary.phase offset of 90 degrees. In this "standing gait Spyndra slowly gyrates the chassis while standing stationary.

The second dataset was recorded as Spyndra performed a stored randomly generated "walking gait," with the pattern shown in Figure 4a. A phase offset of 45 degrees was used between legs. As friction of the walking surface affects Spyndra's gait, it is important to note that the experiment was conducted on a linoleum floor. The first dataset includes data from 8 runs while the second dataset includes data from 14 runs.

The sequence for each leg starts with the Spyndra on a flat surface. The femur

The femur is lowered until it touches the flat surface at its lowest point. The PWM period of this point is recorded

The femur's position is raised twelve degrees from the position of contact.

Now, the tibia is lowered until it touches the flat sur-face at its lowest

Figure 7: The calibration sequence for a single leg

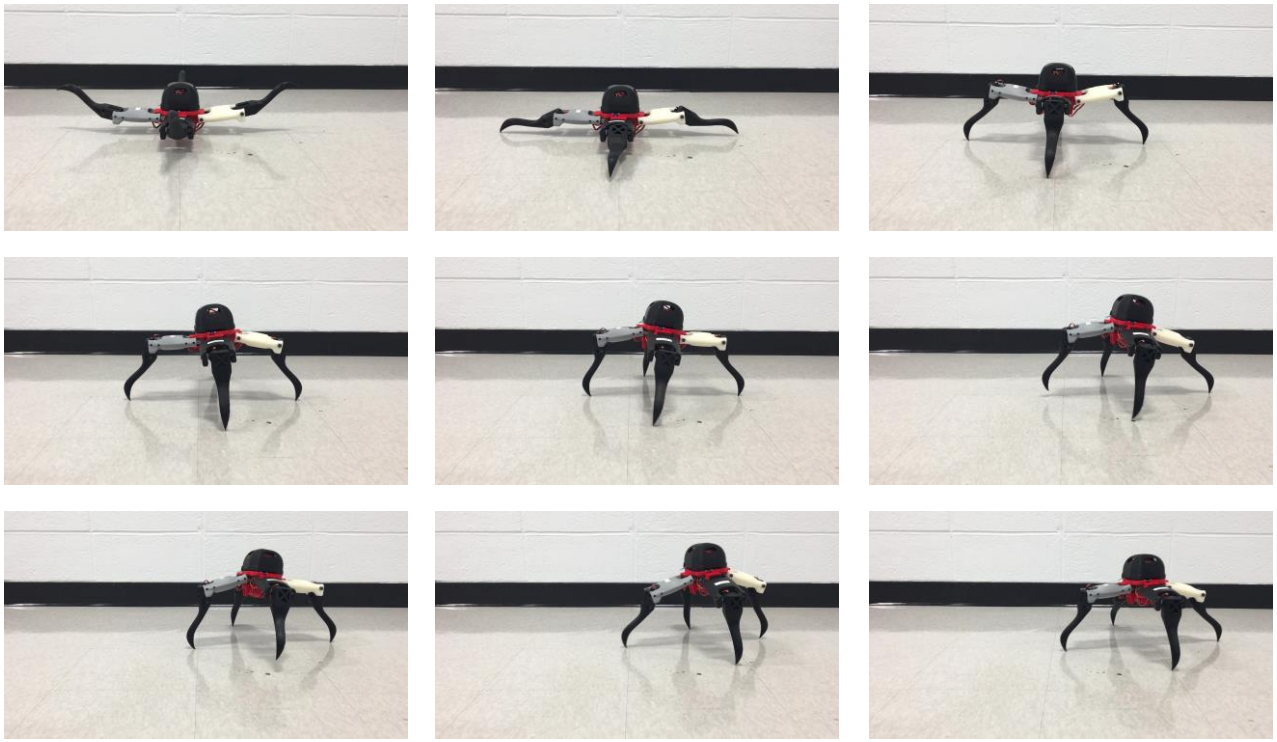
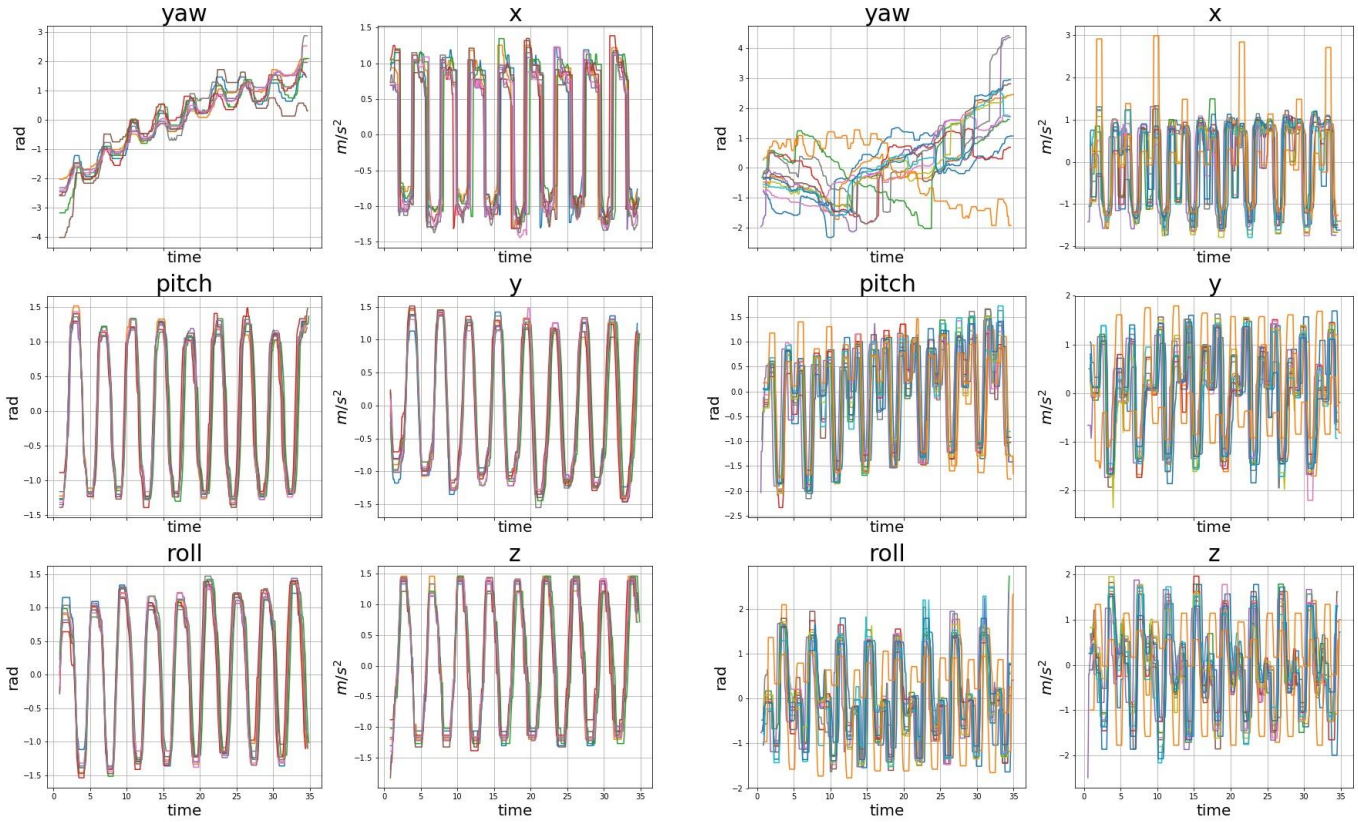


Figure 8: A frame by frame illustration of Spyndra's "walking gait" measured for IMU dataset. Camera position is fixed



(a) Standing

(b) Walking

Figure 9: IMU data plotted against time. Gyroscopic data in degrees, acceleration data in m=s2

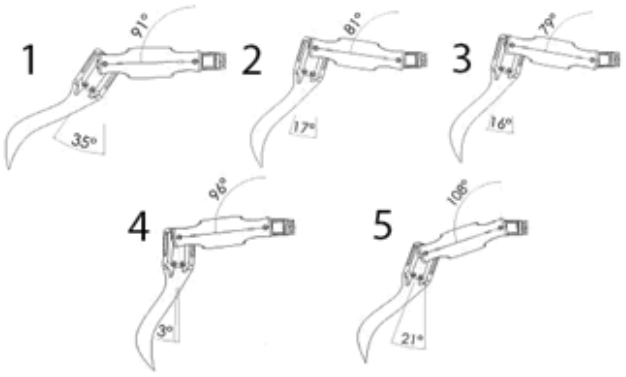


Figure 10: Illustration of the "walking gait" performed by Spyndra for data collection. An offset of 45 degrees was used between legs.

For both experiments, standard hardware was used, and Spyndra's motors were powered by the LiPo battery so it could move untethered. The raw data can be found on the Spyndra website as well as the Python script used to process it. Data processing included filtering of anomalous high and low values, use of a median filter to remove sharp spikes in data, and normalizing of data to account for differences in initial orientation.

	Standing		Walking	
	Correlation	STD	Correlation	STD
Yaw	0.8746	1.3201	0.3896	6.5146
Pitch	0.9664	1.6955	0.9263	3.6387
Roll	0.9582	1.0643	0.8124	1.9879
X	0.8209	0.2881	0.8063	0.7994
Y	0.8746	0.1849	0.8403	0.3191
Z	0.9840	0.0420	0.5062	11.4575

TABLE 4: DISTRIBUTION OF CORRELATION COEFFICIENTS OF IMU DATA ACROSS REPETITIONS

VI. MACHINE LEARNING

In order to achieve self-awareness, we set up two problems: global movement prediction and IMU prediction. The global movement problem aims at predicting the distance, direction and orientation by the 5 random interpolate points for femur and tibia as input features. This problem is important for path planning. Given a high-level route, the robot should be able to generate the appropriate gait to follow such a route.

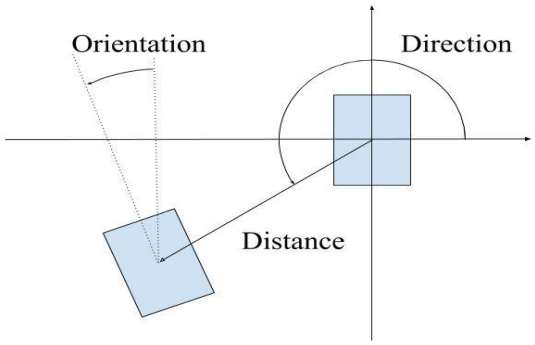


Figure 11: Definition of Global Measurements

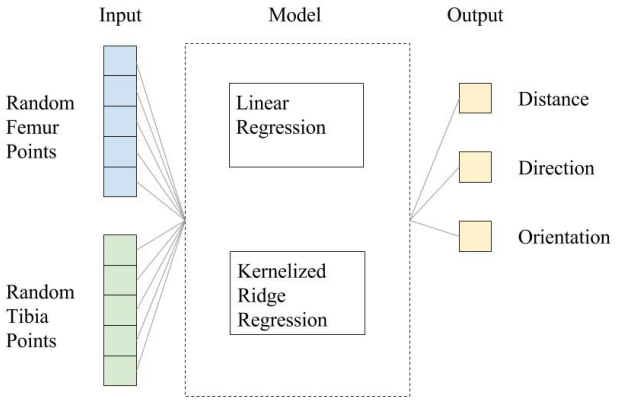


Figure 12: Definition of Global Prediction Problem

	Linear Regression		Ridge Regression	
	Training MSE	Testing MSE	Training MSE	Testing MSE
Distance	353.04	1315.17	371.45	1019.92
Direction	1.31	9.17	1.44	6.62
Orientation	0.0026038	0.031480	0.027071	0.019467

Table 5: Evaluation of Global Prediction

We used two baseline models to predict robot motion: linear regression and ridge regression. The mean square errors of both methods are shown in Table 5.

One example of our prediction testing is shown in Figure 13. Neither the linear regression nor the ridge regression predicted the global measurement precisely. Since the direction and the orientation are related to yaw, and yaw is the least repeatable feature, it is likely that the direction and the orientation are hard to predict as well.

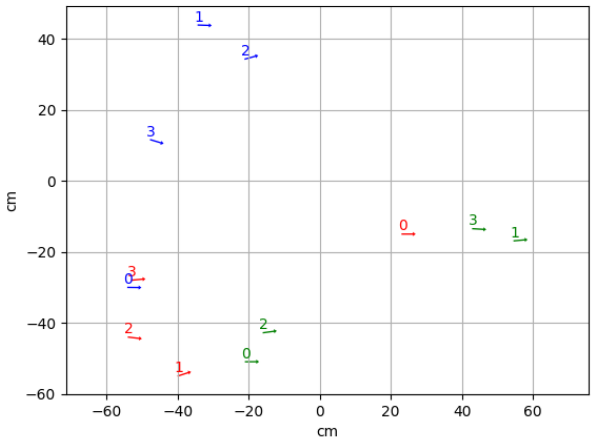


Figure 13: Global Prediction (red: ground truth, blue: linear regression, green: ridge regression)

IMU prediction, on the other hand, deals with the problem of predicting the current state (yaw, pitch and roll) and acceleration based on motor commands and previous IMU

measurements. A motor command consists of 8 motor angles, and an IMU measurement consists of yaw, pitch, roll and x-y-z acceleration. This kind of self-awareness focuses on how the actions of the robot interact with itself and the environment. For example, the trained model should be able to adapt to either a slippery ground or a tilted surface ideally, depending on the motor command (action) and IMU measurement (environmental feedback).

Because a gait is a time-dependent motion, it is likely that the robot's previous time-step is as important as motor commands. Hence, we merged the motor command and IMU measurements from previous the time-steps as input features. The architecture of our neural network is shown in Figure 14.

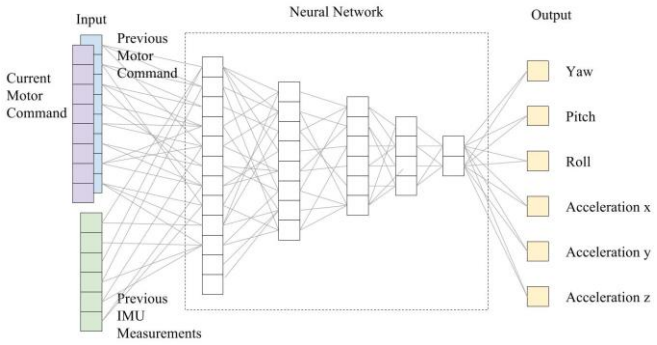


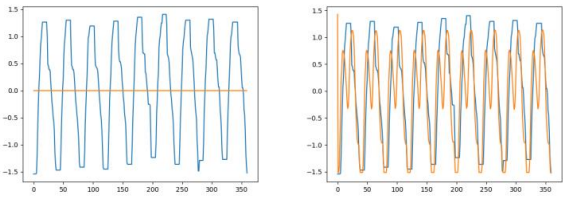
Figure 14: Architecture of fully-connect neural networks

To experiment with different neural networks, 14 random walks were used for training and 4 other random walks were used for testing. We compared the performance of using one, two and three time-steps of information and listed the results in Table 6.

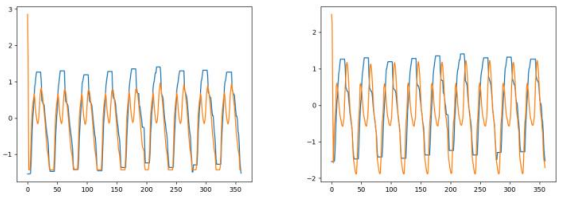
	MSE		Avg. Correlation Coef.	
	Training	Testing	Training	Testing
Current step	1.0538	1.0115	N/A	N/A
1 step	0.2115	0.4334	0.9096	0.8088
2 step	0.1905	0.2691	0.9236	0.8863
3 step	0.1876	0.3384	0.9354	0.8310

Table 6: Evaluation of IMU Prediction

It was observed that the neural networks failed to make meaningful predictions without previous commands and measurements. This fact verifies our assumption that the state of the robot depends on the robot's previous state.



(a) (b)



(c) (d)

Figure 15: Ground truth (blue) and prediction (orange) of pitch of (a) without previous step features (b) 1 step model (c) 2 step model (d) 3 step model.

Although the third order model achieved the lowest mean squared error and highest correlation in training phase, the second order model did better in testing phase. One possibility is that the neural networks began to overfit the training dataset after introducing features from 3 time-steps before. In this case, adding more training data may alleviate the overfitting issue. Another possibility is that 2 time-steps is the optimal feature set for our purpose.

VII. Simulation Model

In order to acquire the best feature set, we need a larger dataset to rule out the possibility of overfitting. We were also aware that data collection is a time-consuming process. This motivates us to build a model that generates IMU measurement based in a simulation environment. With a simulation model, we are able to generate training data quickly and hence accelerate the development process. Plus, we can experiment with different sensors which we do not have in the real world. For instance, the global data was measured manually, because Spyndra does not have any localizing sensor. In ROS, however, we can obtain the position with lines of code.

Our first model was built on Adams, a popular software for calculating vehicle dynamics. However, Adams had some limitations when we applied it to simulate Spyndra. The servo motor of Spyndra has a built-in closed-loop controller to deal with the error. Even if it is able to calculate the acceleration and pose of the robot, it does not account for the error between the actual motion and command. Moreover, its integration to other software is limited. We could only simulate through its graphical user interface, which makes it hard to automate the process of data collection.

Such limitations led us to choose the Robotics Operating System (ROS) and Gazebo as our new simulation tool. ROS provides comprehensive libraries, software and tools for general robot platforms. With ROS, we have every component just like the physical robot. We used its proportional–integral–derivative (PID) controller to mimic the behavior of the servo motors. ROS also provides an IMU sensor plugin with which we can generate IMU measurements.

For simplicity, we modeled Spyndra with simple blocks and cylinders as shown in Figure 8. We also specified the dynamic properties such as moment of inertia and mass. After setting up the robot description, the physics engine of Gazebo simulated the motion of given motor commands.

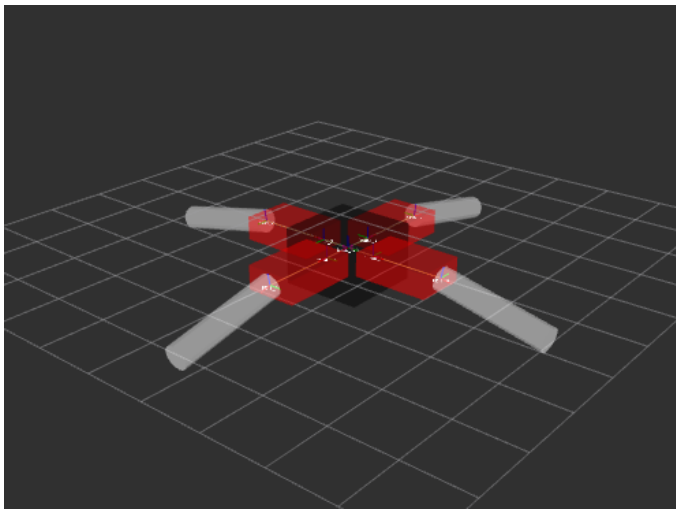


Figure 16: Simplified model description

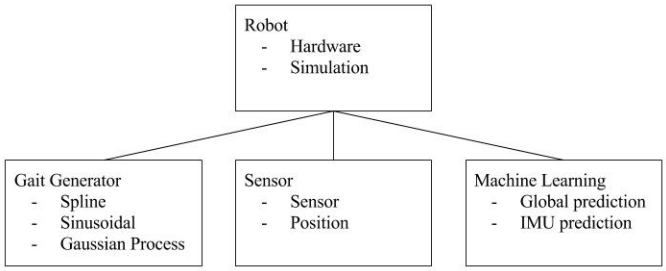
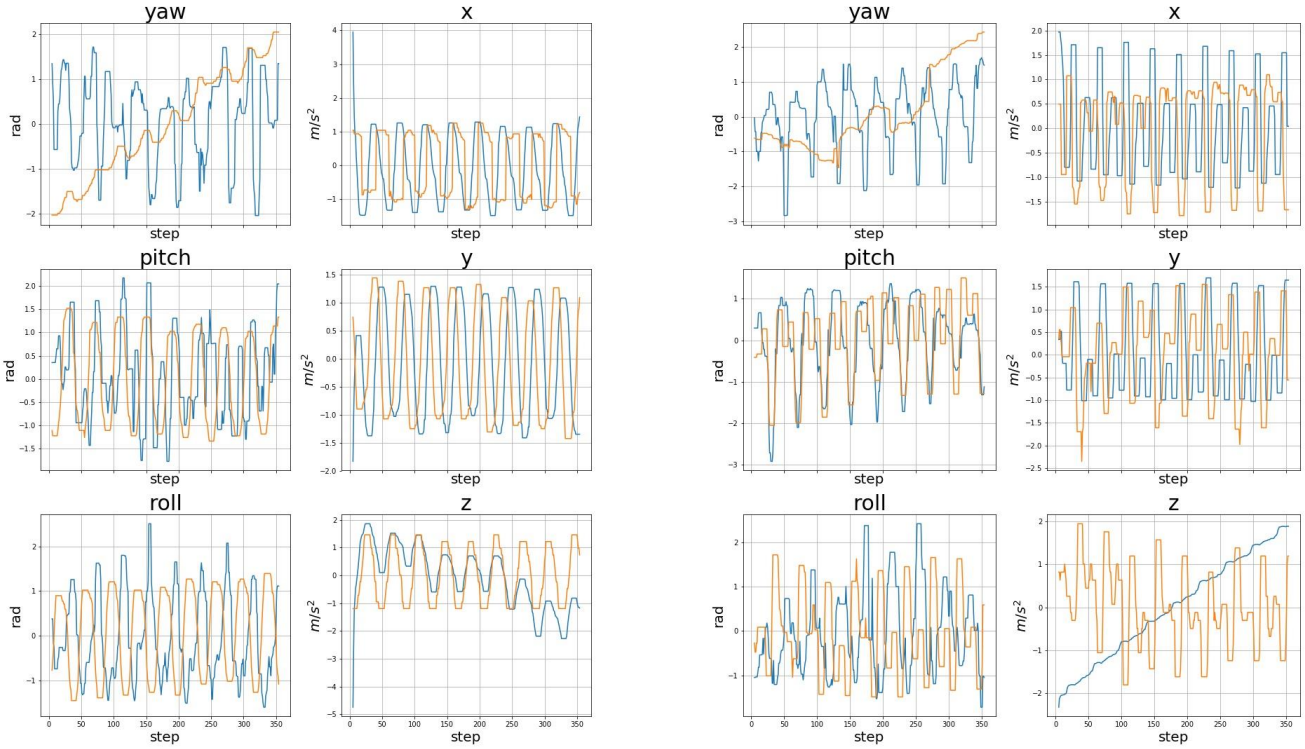


Figure 17 – Software framework of Spyndra

In order to validate the simulated IMU data, we wrote a program that takes in a past walk, simulates it and compares IMU measurements. The normalized data of a simulated standing gait and walking gait are presented in Figure 8. It can be observed that acceleration of x, y direction have a higher correlation, which corresponds to higher repeatability.

There are two reasons for such divergence between the simulation and reality. One reason is that the simulation model assumes that the material is homogeneous, whereas the robot has a hollowed structure due to 3D printing. The center of mass (COF), as a result, might be skewed to a side. Since the walking is highly related to translation of center of mass, it is not surprising such skewed COF affects the pose of Spyndra. Another factor is the quality of the physics engine. During the simulation, the legs of the model were sometimes shaking because the physics engine did not handle the computation well. This introduced extra perturbation to the simulated IMU data.

Even though the simulation has noticeable differences to the real data, it is an effective way of generating various training data. The gait does not necessarily have to be spline functions. It can be either simple sinusoidal waves or Gaussian processes. The simulation model allows us to parallel the processes of collecting data on the physical robot and develop machine learning algorithm. Therefore, we wrote a programming interface to ensure that the application applies to both simulation and physical experiments. The planned software framework is shown in Figure 17.



(a) (b)

Figure 18: ground truth (blue) and simulation (orange) of (a) standing gait (b) walking gait

VIII. Conclusion

We have introduced Spyndra: an open source quadruped robot meant to serve as a platform for robotics and AI re-searchers interested in self-awareness. Comprised of 3D-printed parts and off-the-shelf hardware, Spyndra is inexpensive, easy to assemble, yet achieves complex kinematics. We have also constructed a few basic machine learning models to begin allowing Spyndra to create its self-model; these machine learning models are also compatible with both Spyndra's hardware and the simulated version for better data acquisition. To aid in the implementation of machine learning for others, open source control software, a set of baseline IMU data, machine learning code, and simulation files are available for future researchers. These materials make Spyndra ideal for hardware implementation of machine learning and self-modelling software, and we hope it will serve as a common starting point among roboticists, academics, and the broader AI community.

In the future, we would like to strengthen Spyndra's capabilities as a self-modeling platform by increasing the feed-back sensors on board using the features we extract from our simulation. With additional proprioceptive sensors, Spyndra can learn more about itself. In addition, we will collect more data from the IMU, growing the publicly available dataset, and continue enhancing our current machine learning models.

IX. Acknowledgement

This work has been supported in part by a gift from Northrop Grumman Incorporated, for the study of self-aware systems.

References

[1] Bongard J., Zykov V, Lipson H. (2006). "Resilient machines through continuous self-modeling." Science, 314(5802):1118-21.

[2] Hornby, G., Takamura, S., Yamamoto, T., and Fu-jita, M. (2005). Autonomous evolution of dynamic gaits with two quadruped robots. IEEE Transactions on Robotics, 21(3):402410.

[3] Koza, J. (2003). Genetic programming IV: Routine hu-man competitive machine intelligence. Kluwer.

[4] Lipson, H. and Pollack, J. (2000). Automatic de-sign and manufacture of robotic lifeforms. Nature, 406(6799):974978.

[5] Lohmann, Sara, Jason Yosinski, Eric Gold, Jeff Clune, Jeremy Blum, and Hod Lipson. "Aracna: An Open-Source Quadruped Platform for Evolutionary Robotics." Artificial Life 13 (2012): n. pag. Web.

[6] Mnih, V. et al. Human-level control through deep rein-forcement learning. Nature 518, 529533 (2015).

[7] Nolfi, S. and Floreano, D. (2000). Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines. MIT Press, Cambridge, MA.Pfeifer, R., Bongard, J., and Grand, S. (2007). How the body shapes the way we think: a new view of intelli-gence. The MIT Press.

[8] Pfeifer, R., Bongard, J., and Grand, S. (2007). How the body shapes the way we think: a new view of intelli-gence. The MIT Press.

[9] Standard, A. S. T. M. "F2792. 2012 Standard ter-minology for additive manufacturing technologies." West Conshohocken, PA: ASTM International. doi: 10.1520/F2792-12

[10] Sims, K. (1994). Evolving 3D morphology and behavior by competition. Artificial Life, 1(4):353-372

[11] Yosinski J., Clune, J., Hidalgo, D., Nguyen, S., Zagal, J., and Lipson, H. (2011). Evolving robot gaits in hard-ware: the hyperneat generative encoding vs. parameter optimization. In Proceedings of the European Confer-ence on Artificial Life, pages 890-897

[12] Spyndra [Online] Creative Machines Lab Columbia University [Online]. Available http://www.creativemachineslab.com/spyndra.html [7 Apr 2017]

[13] Hend, D Arduino Quadruped Robot Instructables. [Online]. Available: http://www.instructables.com/id/Symnopsis/[7 Apr. 2017]

[14] Lynxmotion SQ3U Walking Robot Lynxmotion. [Online]. Available: http://www.lynxmotion.com/c-26-quadrapods.aspx [7 Apr. 2017]

[15] Hsu, R[DIY] Spider Robot(Quad Robot, Quadruped) Instructables [Online] http://www.instructables.com/id/DIY-Spider-RobotQuad-robot-Quadruped/[7 Apr. 2017]

[16] Hatfield, S.A 3D Printed Quadruped Robot Instructa-bles. [Online]. http://www.instructables.com/id/A-3D-Printed-Quadruped-Robot/[7 Apr. 2017]

[17] AracnaCreativeMachinesLab-Columbia University. [Online].Available: http://www.creativemachineslab.com/aracna-robot.html [7 Apr. 2017]

[18] Bongard, J. Starfish Mechanical and Aerospace Engineering - Cornell University. [Online]. Available: http://www.pnas.org/content/108/4/1234.abstract [7 Apr. 2017]

[19] Lynxmotion A-Pot Hexapod Lynxmotion [Online]. Available: http://www.robotshop.com/en/lynxmotion-a-pod-hexapod-robot-kit-no-electronics.htmlSpecifications [15 Apr. 2017]

[20] Hexy-Programmable Hexapod Kit Adafruit. [Online]. Available: https://www.adafruit.com/product/1529 [15 Apr. 2017]